DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 2: “Análisis predictivo de símbolos numéricos en imágenes mediante modelos predictivos”.

Laboratorio de Datos.

Grupo “Pescado Rabioso”

Integrante	LU	Correo electrónico
Souto, Sebastian Manuel	43/23	soutosebastianma@gmail.com
Sanza, Gian Lucca	149/23	gianluccalord723@gmail.com
Goldfarb, Bruno	1164/23	bgoldfarb2003@gmail.com

1. Resumen

Este trabajo tiene como objetivo desarrollar y analizar diferentes modelos predictivos basados en una base de datos que contiene imágenes de dígitos del 0 al 9, representados en diversas fuentes. Se abordará tanto la clasificación binaria, seleccionando dos dígitos que a simple vista parecen fácilmente diferenciables, como la clasificación multiclase. En la parte de clasificación binaria, se explorará cómo ciertos atributos distintivos de cada dígito permiten construir modelos efectivos con un número reducido de atributos, pero con métricas de rendimiento altas. En cuanto a la clasificación multiclase, se desarrollará un modelo que sea capaz de predecir el dígito correspondiente a una nueva imagen, independientemente de la fuente tipográfica en la que se presente.

2. Introducción

Comenzando el recuento de datos nos encontramos con un archivo csv que vemos , una vez convertido a un DataFrame, contiene 29900 filas que corresponden a 10 números en 2990 tipografías distintas.
Cada fila está compuesta por el nombre de la tipografía y el número que vamos a dibujar seguido por 784 (28x28) valores que oscilan entre 0 y 255 que corresponden al grado de intensidad que tendrá cada píxel de la imagen.
Trabajar con imágenes agrega complejidad a la exploración de datos. Para empezar recién podemos comprobar que es exactamente lo que estamos viendo una vez ejecutado el código que visualiza la imágen.
Además de eso, por ejemplo, con el dataset del titanic si hubiera un error en la carga de datos que genere una incongruencia lo podríamos notar con relativa facilidad. Por ejemplo alguien con más de 100 años de edad, que puede ser porque le agregaron un 0 atrás, u otro tipo de incongruencias.
Pero en datasets de imágenes, encontrar errores de este estilo requiere analizar caso por caso cada imágen y muchas veces ni siquiera se podría saber si es efectivamente un error, por ejemplo si hubiera una confusión en el valor de un pixel puntual, esto sólo cambiaría literalmente la tonalidad de un gris, haciendo la tarea de reconocer un error ardua.

Para ver si existen atributos que sean relevantes y otros que no vamos a superponer n cantidad de imágenes aleatorias promediando el valor en cada posición.

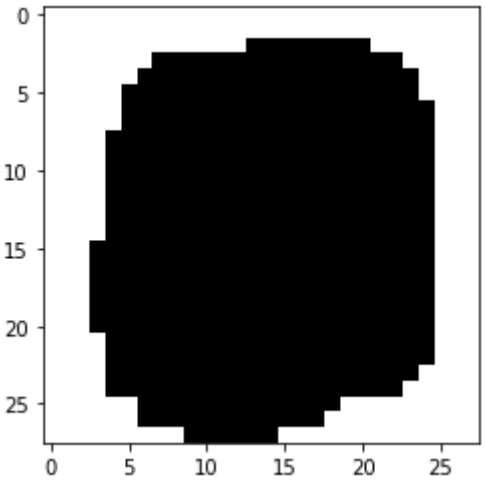

Figura 1. Corresponde al promedio de 1000 imágenes. Usamos binary como mapa de color.

Decidimos cambiar el mapa de colores a uno que permite distinguir mejor los atributos: aquellos con un promedio igual a 0 se muestran en blanco, mientras que los que tienen un promedio mayor aparecen en negro. Se aprecia a simple vista que todos los atributos correspondientes a las primeras y las últimas 3 al igual que las primeras 2 filas son descartables por tener un promedio igual a 0, es decir que de las 1000 imágenes ninguna usaba esos pixeles.

Con el fin de averiguar si existen distintos números que sean más fáciles de distinguir entre sí decidimos promediar los valores correspondientes solo a ciertos números.

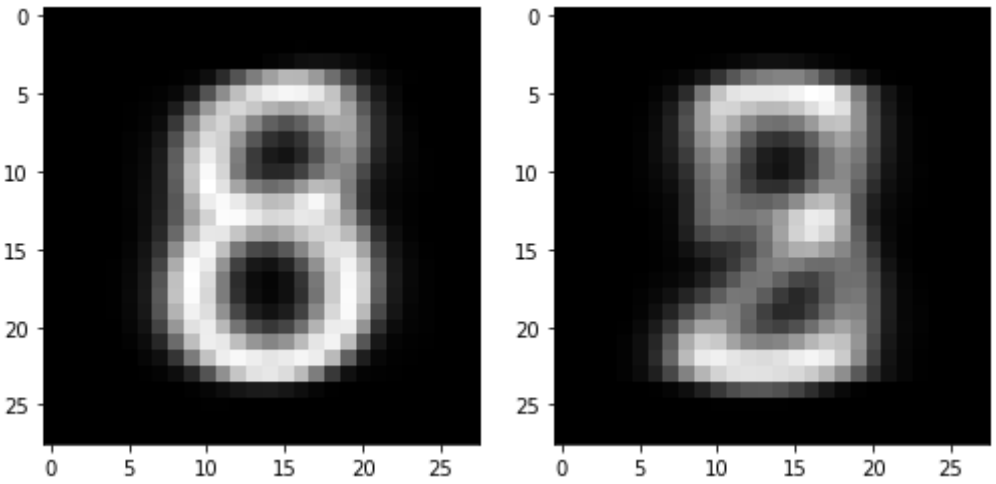

Figura 2. Corresponde a 2 imágenes: a la izquierda, el promedio de 500 imágenes de los números 6 y 8; a la derecha, el de 500 imágenes de los números 2 y 5

Podemos ver con mucha claridad que el 8 y 6 están mayormente superpuestos. En cambio cuando promediamos los dos y los cincos se pueden distinguir claramente ambos números. Podemos afirmar que la diferencia entre distintos números depende de qué número estemos comparando. También nos resulta interesante analizar qué tanto se parecen entre sí cada número. Para eso, vamos a hacer las pruebas con el número 0:

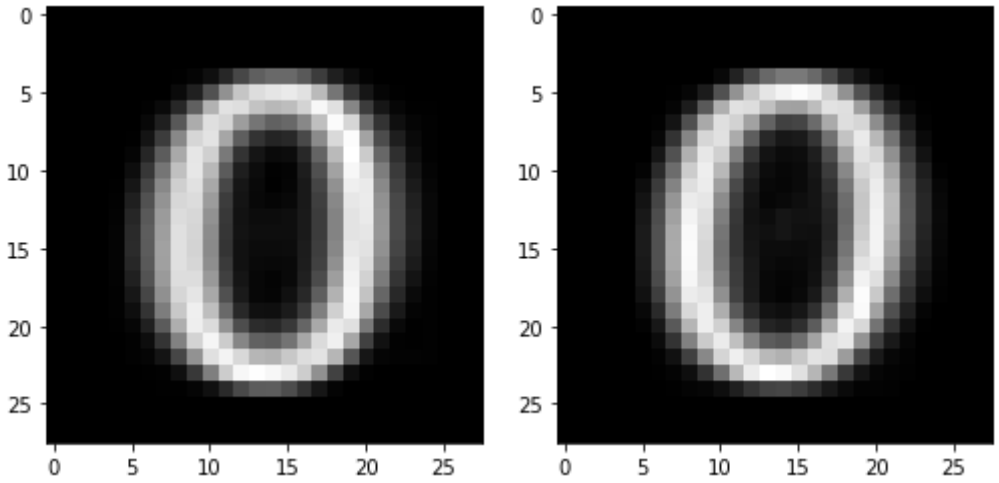

Figura 3. Está compuesta por dos imágenes resultantes de correr el mismo código dos veces. Este código elige aleatoriamente 100 imágenes correspondientes de 0 y promedia sus atributos.

Vemos que aunque no son exactamente iguales, son muy similares y hay que prestar mucha atención para notar las diferencias. Por la similitud entre los dos promedios (recordemos que se tratan de 100 imágenes dentro de 2990) podríamos llegar a la conclusión de que las imágenes de distintas tipografías de un mismo número son muy parecidas entre sí.

Verificamos cómo se ven bajando significativamente la cantidad de imágenes:

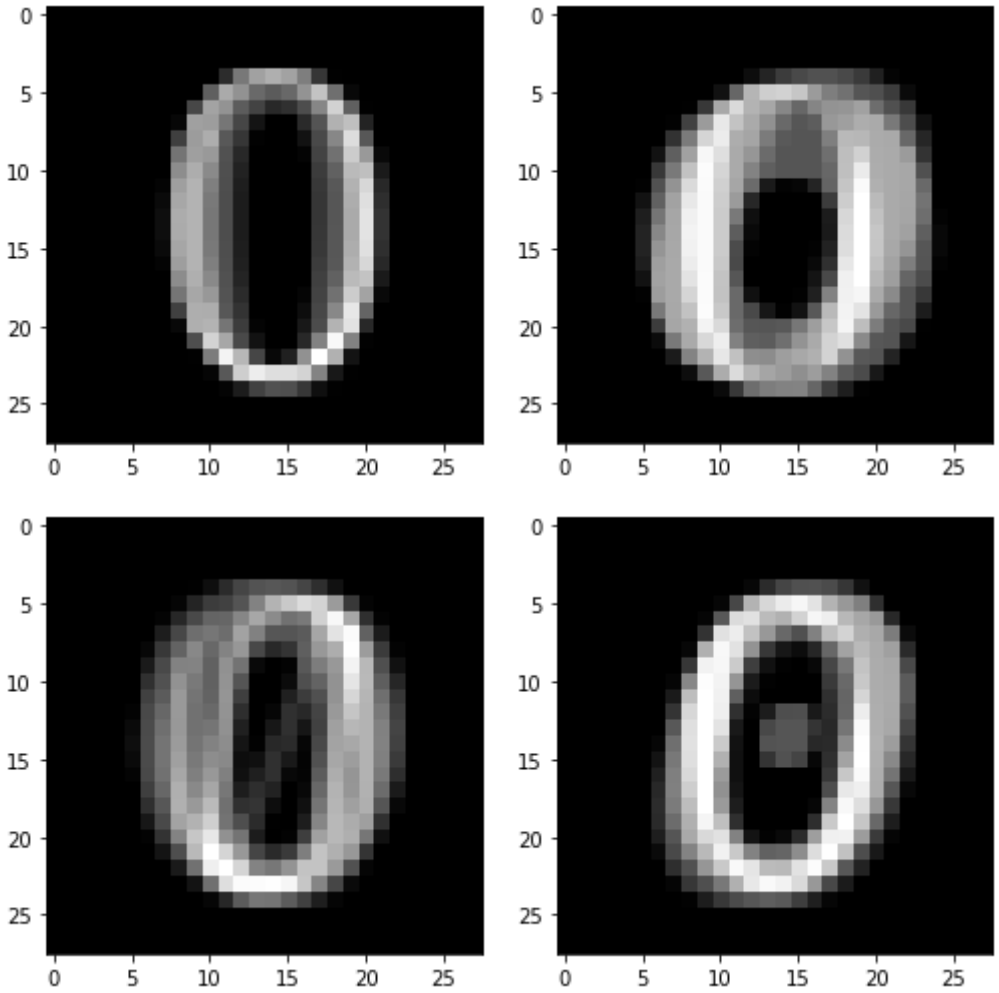

Figura 4. Está compuesta por dos imágenes resultantes de correr el mismo código dos veces. Este código elige aleatoriamente 3 imágenes correspondientes de 0 y promedia sus atributos.

Al bajar considerablemente la cantidad de imágenes promediadas podemos ver que las diferencias son ahora clarísimas. Tienen distinto grosor, solidez, altura y algunos incluso tienen relleno.
Si comparamos individualmente cada imágen vamos a notar las diferencias, pero todas comparten similitudes que son las que apreciamos al promediar grandes cantidades.
Para responder las preguntas correspondientes vamos a utilizar distintos tipos de modelos predictivos. Para cada caso se analizaran las métricas en función de los hiperparametros correspondientes a cada modelo en busca del mejor modelo en términos de rendimiento. Esto último podrá ser comprobado midiendo el tiempo que se toma el programa en ejecutar las órdenes, y con las métricas proporcionadas por la teoría de modelos (Precisión, Exactitud, entre otras).

3. Desarrollo Experimental.

En primera instancia se desarrollaron distintos modelos de KNN sobre el conjunto de datos filtrado por ceros y unos. Se quiso observar cuales son los atributos que pueden diferenciar a estos dos dígitos y como afecta la variación de los hiperparametros en las métricas de los modelos.

Primero se fijó la cantidad de vecinos en 5 y se fue variando sobre qué atributos tomar para hacer el modelo. Para las métricas se tomó al 1 como la clase positiva. Se obtuvieron los siguientes resultados:

Tabla 1. Distintos modelos de KNN con sus respectivas métricas

Atributos	**Exactitud**	**Precisión**	**Recall**
Todos los atributos	0.996	1.0	0.992
Todos menos los bordes	0.996	1.0	0.993
3 atributos al azar	0.864	0.847	0.891
3 atributos no vacíos.	0.887	0.960	0.803
1 atributo del medio	0.857	0.911	0.796
3 atributos del medio.	0.933	0.933	0.933
7 atributos del medio.	0.975	0.981	0.971

Primero se desarrolló un modelo con todos los atributos. Las métricas fueron muy buenas, pero como se mencionó anteriormente hay muchos atributos que se pueden descartar. Luego se utilizaron todos los atributos menos los bordes, descartando un total de 180 atributos. Las métricas nuevamente fueron muy buenas.
Sin embargo, estamos comparando dos dígitos que se diferencian fácilmente entre sí. Para mostrar esto construimos la siguiente imágen, que está construida por el promedio de 500 imágenes de ceros y unos.

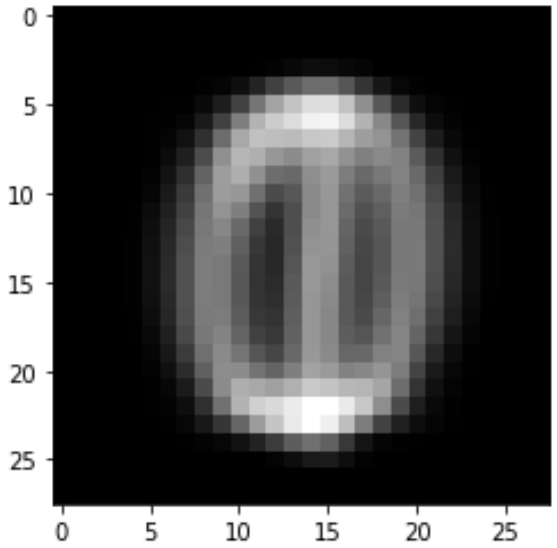

Figura 5. Promedio entre imágenes de ceros y unos.

Como se observa en la Figura , se pueden distinguir los dos dígitos. Esto se debe a que los atributos que corresponden a los píxeles del centro de las imágenes tienen valor 0 para los ceros, mientras que para los unos no.

Para confirmar esta idea se desarrollaron diferentes modelos con 3 atributos. El primero fue con 3 atributos tomados al azar. En este modelo las métricas disminuyeron con respecto a los modelos desarrollados anteriormente, por lo que se siguió con la búsqueda de 3 atributos representativos. Luego se tomaron 3 atributos que se sabe que tienen información para todos los dígitos, es decir que su valor es distinto de 0 en la mayoría de los casos. En este caso las métricas aumentaron, pero no se está aprovechando del todo las diferencias entre los dígitos dados. Por eso, usando lo mostrado en la Figura, se tomó un atributo correspondiente al centro de las imágenes. Este modelo desarrollado con 1 solo atributo obtuvo métricas similares a los modelos desarrollados anteriormente con 3 atributos. Finalmente se tomaron 3 atributos que están en el centro de las imágenes para desarrollar un modelo. Este mismo aumentó las métricas con respecto a los demás modelos desarrollados con 3 atributos, confirmando así la idea de que los atributos que se encuentran en el centro de las imágenes son los más representativos para diferenciar los ceros de los unos.

Teniendo en cuenta lo anterior, se desarrolló un modelo utilizando 7 atributos correspondientes a los píxeles centrales de las imágenes, lo que resultó en una mejora significativa en las métricas, acercándose a los resultados obtenidos por los modelos que utilizaron todos los atributos.

Una vez que se eligieron los 7 atributos a utilizar para desarrollar el modelo, se varió la cantidad de vecinos para observar cómo esto afecta a las métricas. Se utilizaron vecinos en el rango de 5 a 800. Y se obtuvieron los siguientes resultados:

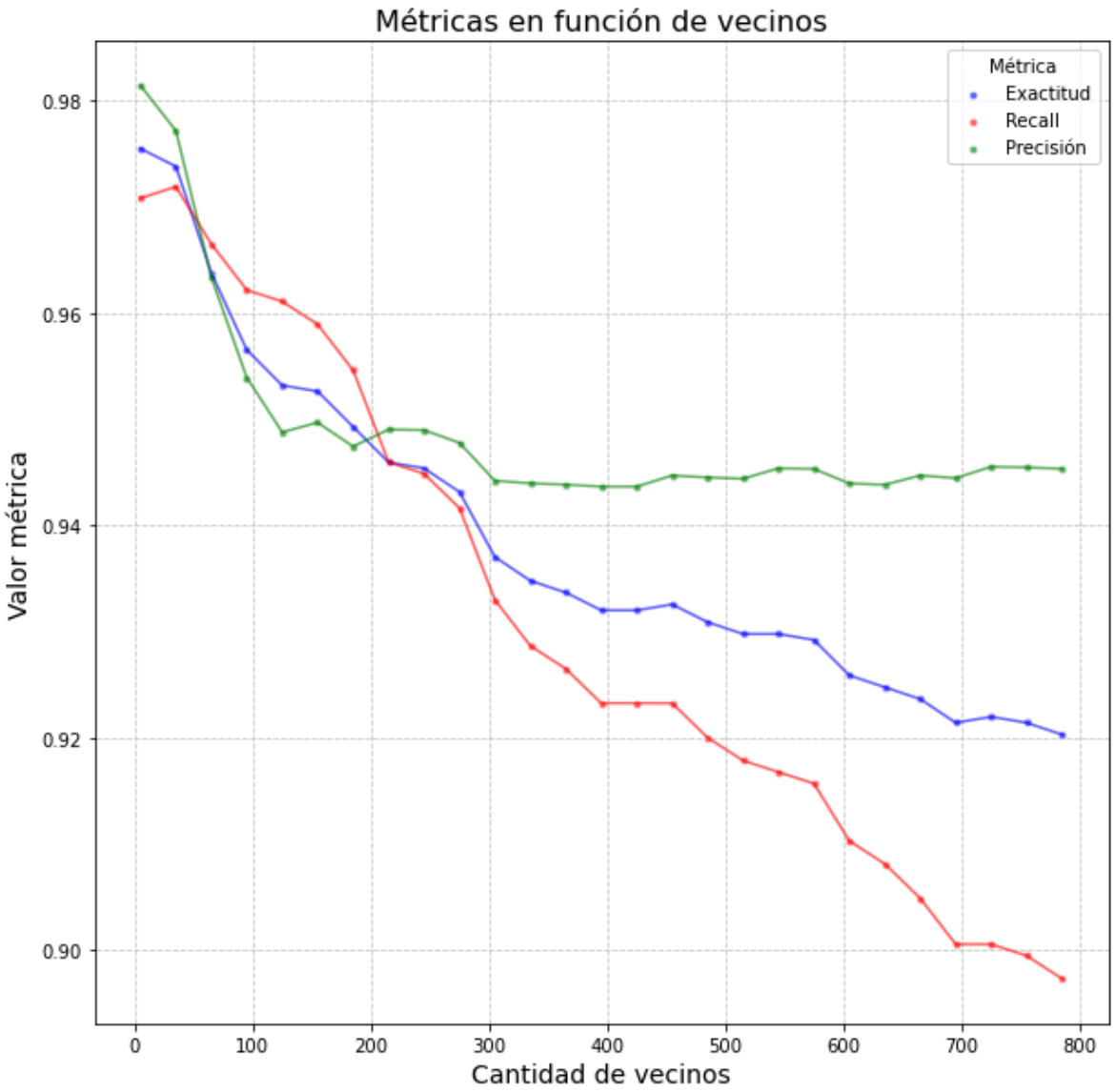

Figura 6. Valores de las métricas en función de la cantidad de vecinos.

Respecto a la exactitud y recall se puede observar que a medida que aumenta la cantidad de vecinos estas métricas tienden a disminuir. Mientras que la precisión primero disminuye y luego se mantiene. De esta forma ninguna de las 3 métricas vuelve a alcanzar el valor obtenido al utilizar el valor inicial. Por esto, se puede pensar que aumentar el valor de los vecinos, en este caso, no mejora el modelo desarrollado.

Una vez analizado el conjunto de números separados binariamente, se hizo un análisis de clasificación multiclase. Para esto se optó por utilizar modelos de árboles de decisión. Primero se separaron los datos en desarrollo y validación, y luego se evaluó qué profundidad de análisis rendía más para nuestro propósito.

En un entorno de desarrollo se evaluaron 3 modelos utilizando Entropy los datos:

Tabla 2. Exactitud, Recall y Precisión variando la cantidad de depth.

Depth	1	5	10
Exactitud	0.195	0.836	0.928
Recall	0.195	0.836	0.928
Precisión	0.0396	0.842	0.93

Observamos que los datos de recall y exactitud coinciden en todos los casos. Teniendo en cuenta la teoría, esto debería significar que para cada caso, el número de FP es igual al número de FN.
Luego, se evaluaron 50 modelos más que variaba en profundidad y modelado (Gini o Entropía) siguiendo la metodología del K-fold cross validation. Se graficaron los resultados de sus respectivas métricas:

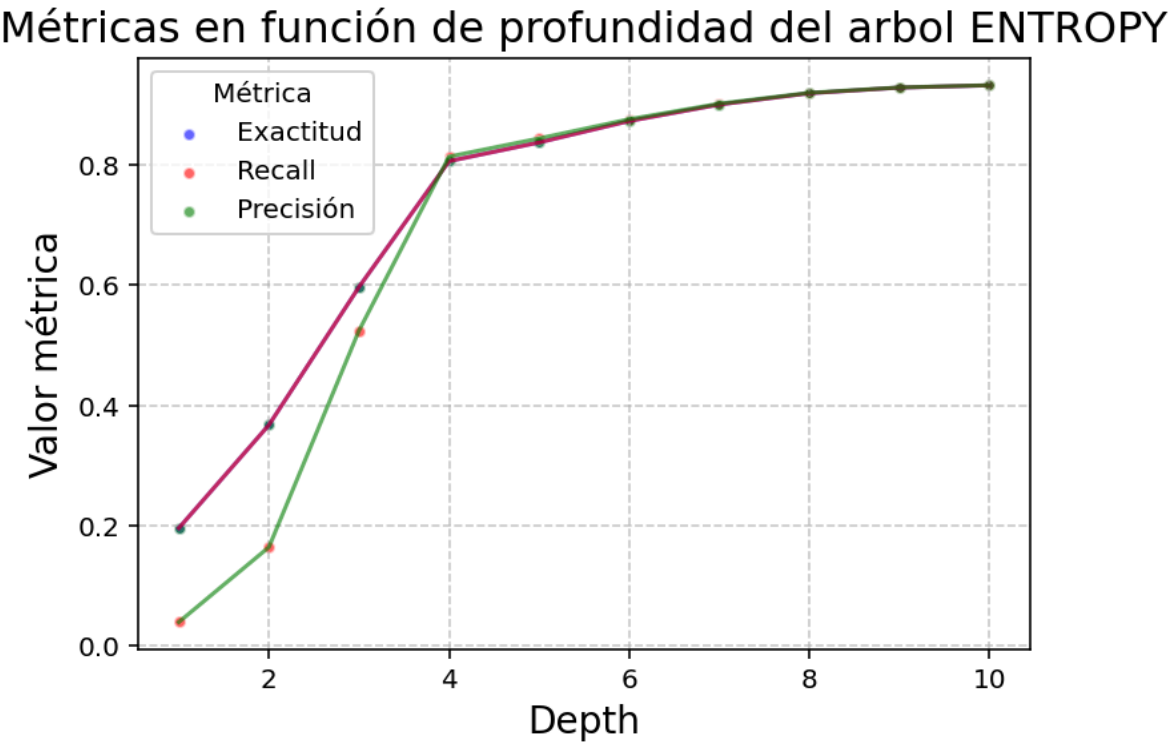

Figura 7. Valores de las métricas en función de la cantidad de depth ENTROPY.

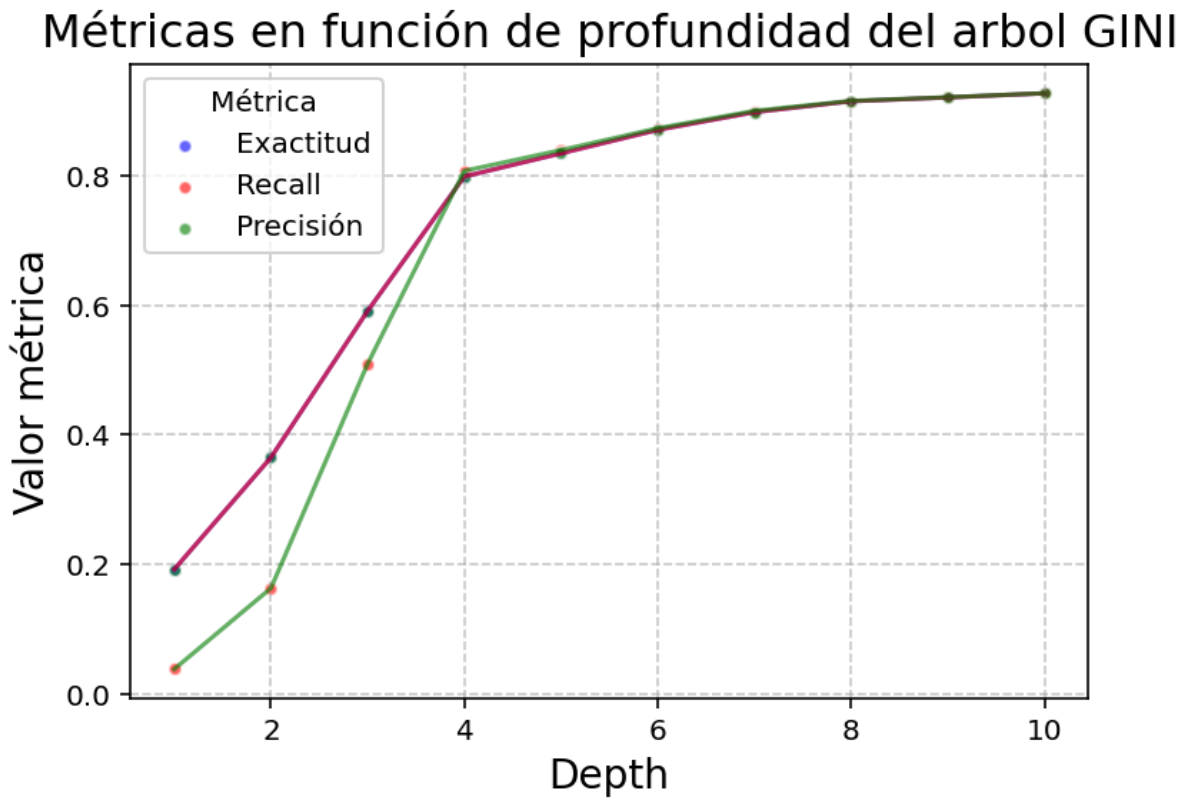

Figura 8. Valores de las métricas en función de la cantidad depth GINI.

Podemos observar que las métricas tienden a un 0.9 a medida que se aumenta la profundidad del árbol. Cómo utilizar una profundidad mayor a 9 ralentiza gravemente el programa, optamos por utilizar una profundidad 9, que tiene métricas satisfactorias para el objetivo del trabajo. Además, el modelado Gini no tiene diferencias significativas con el modelado Entropy. Es por eso que decidimos utilizar Gini como modelo final.

Para confirmar que el modelo rinde con altas métricas, lo evaluamos en el conjunto de datos de validación previamente separado

	Exactitud	Precisión	Recall
Modelo Gini Depth 9	0.957	0.957	0.957

Cuya matriz de confusión se ve así:

$$\begin{bmatrix} 583 & 1 & 0 & 0 & 0 & 0 & 3 & 0 & 4 & 5 \\ 2 & 615 & 0 & 1 & 0 & 2 & 1 & 2 & 0 & 2 \\ 4 & 3 & 545 & 3 & 1 & 1 & 2 & 3 & 3 & 5 \\ 6 & 6 & 1 & 551 & 1 & 8 & 4 & 1 & 5 & 7 \\ 3 & 4 & 0 & 1 & 557 & 2 & 1 & 0 & 1 & 7 \\ 3 & 0 & 1 & 5 & 5 & 578 & 8 & 1 & 7 & 13 \\ 3 & 0 & 0 & 1 & 5 & 2 & 571 & 0 & 7 & 6 \\ 2 & 12 & 0 & 2 & 3 & 2 & 0 & 587 & 0 & 2 \\ 0 & 3 & 1 & 8 & 0 & 6 & 12 & 0 & 561 & 5 \\ 2 & 1 & 0 & 7 & 5 & 6 & 2 & 2 & 7 & 569 \end{bmatrix}$$

4. Conclusiones.

Con respecto a los modelos de KNN desarrollados con los dígitos 0 y 1 podemos concluir que estos dígitos se pueden diferenciar con pocos atributos. Ya que se realizó un modelo con 7 atributos que obtuvo los siguientes valores en las métricas:

Tabla 3. Métricas para el modelo de KNN con 7 atributos del medio.

Exactitud	Precisión	Recall
0.975	0.981	0.971

Afirmamos esta idea debido a que el total de atributos es 784 y se logra alcanzar valores relativamente altos en las métricas con tan solo el 0.9% (aproximadamente) de los atributos.

También podemos concluir que, al aumentar la cantidad de vecinos, el modelo desarrollado no muestra mejoras. La exactitud disminuye de manera notable, lo que significa que en el conjunto de test el modelo comete más errores al hacer las predicciones.

En cuanto a los árboles de decisión, observamos que a medida que aumentamos la profundidad, mejores eran las métricas del modelo. Sin embargo, a partir del depth 7 - 8 las métricas se estancaron, por lo que pedir un modelo que depth mayor a 9 resultaría en desperdiciar recursos. Además, el modelo resultante terminó alcanzando métricas satisfactorias, rondando el 0.9 para cada una de ellas en entorno de desarrollo como de validación.

El recall resultó exactamente igual a la exactitud en cada caso. Sin embargo, no sabemos exactamente cómo es que Sklearn realiza los cálculos cuando se trata de un problema de multiclases. Por lo cual no podemos confirmar que, siguiendo nuestra teoría, la cantidad de FP es igual a la de FN en todos los casos.